Package ‘RelimpPCR’

June 1, 2023

Type Package

Title Relative Importance PCA Regression

Version 0.3.0

Date 2023-05-23

Author Michael Hernandez <micahel@hernandez.ai>, Yuri Balasanov

Maintainer Michael Hernandez <michael@hernandez.ai>

Description Performs Principal Components Analysis (also known as PCA) dimensionality reduction in the context of a linear regression. In most cases, PCA dimensionality reduction is performed independent of the response variable for a regression. This captures the majority of the variance of the model's predictors, but may not actually be the optimal dimensionality reduction solution for a regression against the response variable. An alternative method, optimized for a regression against the response variable, is to use both PCA and a relative importance measure. This package applies PCA to a given data frame of predictors, and then calculates the relative importance of each PCA factor against the response variable. It outputs ordered factors that are optimized for model fit. By performing dimensionality reduction with this method, an individual can achieve a the same r-squared value as performing just PCA, but with fewer PCA factors. References: Yuri Balasanov (2017) <https://ilykei.com>.

Depends R (>= 3.3.0)

Suggests parallel, testthat

License MIT + file LICENSE

URL https://github.com/mhernan88/RelimpPCR

BugReports https://github.com/mhernan88/RelimpPCR/issues

Encoding UTF-8

RoxygenNote 7.2.3

Imports relaimpo, Rmisc, caret, ggplot2, reshape2, logger

NeedsCompilation no

Repository CRAN

Date/Publication 2023-06-01 16:40:02 UTC

R topics documented:

`RelimpPCR` *A Relative Importance PCA Regression Function*

Description

This function performs a relative importance PCA regression. It performs PCA and then applys a relative importnace measure on each additional factor. The output shows optimal PCA factor selection for a given regression.

Usage

```
RelimpPCR(
  Y,
  X,
  target_r2,
  validation_split = 1,
  relimp_algorithm = "last",
  max_predictors = 0,
  remove_factors = TRUE,
  factors_to_remove = 0,
  max_factors_to_remove = 15,
  normalize_data = TRUE,
  plot_this = TRUE,
  verbose = FALSE,
  multicore = TRUE,
  cores = 2,
  random_seed = NA
)
```

Arguments

`Y`	(list/vector): This a list/vector of Y values for the regression.
`X`	(data frame): This is the input data for the regression.
`target_r2`	(float 0-1): The algorithm will attempt to return to you the simplest model (i.e. with fewest predictors) that satisfies your target_r2 value; If no model satisfies this condition, then the full model (with all predictors) will be returned.
`validation_split`	(float 0-1): This determines how much of your data set will be in the train data set. The remainder will be allocated to the test data set. If set to 1, train and test samples will be identical.

`relimp_algorithm`	(string): This is the "type" of relative importance that will be used for measuring raw predictors (not PCA factors).
`max_predictors`	(int): The maximum number of predictors/factors you want reviewed. Note: For importance measures all predictors/factors will be analyzed for relative importance. Rather, this limits how many predictors/factors are added onto the model to show iteratively increasing R-Suared.
`remove_factors`	(bool): If any eigenvalue, resulting from performing PCA on your data set, is too small for relative importance, it can be removed automatically if this is TRUE. If FALSE, the same situation will produce an error.
`factors_to_remove`	(int): If remove_factors is TRUE, you can either a) set this to 0 to have the script iteratively remove PCA factors until the relative importance calculation works (recommended if you do not know how many PCA factors to remove, but takes longer), or b) set this to any positive integer smaller than the number of factors. In condition b, the script will go ahead and remove the X smallest factors (X being the number this argument is set to).
`max_factors_to_remove`	(int): If remove_factors is TRUE and factors_to_remove is 0, then this will determine how many factors the script will delete before "giving up". This is to prevent a possible very long process. This can be set to 0 to iterate through all columns (not recommended).
`normalize_data`	(bool): Whether or not to normalize (subtract mean and divide by standard deviation) before analysis.
`plot_this`	(bool): Whether or not to plot the r-squared values. Default is TRUE.
`verbose`	(bool): Whether or not to include some additional narration around the status of the process. Default is FALSE.
`multicore`	(bool): Whether or not to use mclapply instead of sapply. Default is TRUE.
`cores`	(int): The number of cores to distribute work across for multicore operations.
`random_seed`	(int): Random seed (if you wish to use one). NA indicates no random seed.

Value

out (list): A list containing all of the below components...

$pca_loadings: The PCA loadings.

$pca_object: The trained PCA object.

$pca_factors_rank: The numerical ranking of the PCA factors.

$original_r2_train: The r-squared values when iteratively adding unordered training predictors.

$pca_r2_train: The r-squared values when iteratively adding unordered training PCA factors.

$relimp_pca_r2_train: The r-squared values when iteratively adding ordered training PCA factors (ordered by relative importance of the training data set).

$best_model: The model with the fewest predictors that has r-squared equal to or above the "target_r2" argument.

$num_factors: The number of PCA factors used in the best model.

$scaling_factors: The mean and standard deviations used to scale the X columns and Y column.

$relimp_r2_train: ONLY RETURNED IF relative importance for ordered predictors is successful. This contains the r-squared values when iteratively adding ordered predictors (ordered by relative importance of the training data set).

$ranked_features: ONLY RETURNED IF relative importance for ordered predictors is successful. This contains the numerical ranking of predictors.

$original_r2_test: ONLY RETURNED IF validation_split argument is not equal to 1. This contains the r-squared values when iteratively adding unordered testing predictors.

$pca_r2_test: ONLY RETURNED IF validation_split argument is not equal to 1: This contains the r-squared values when iteratively adding unordered testing PCA factors.

$relimp_pca_r2_test: ONLY RETURNED IF validation_split argument is not equal to 1. This contains the r-squared values when iteratively adding ordered testing PCA factors (ordered by relative importance of the training data set).

$relimp_r2_test: ONLY RETURNED IF validation_split argument is not equal to 1 AND relative importance for ordered predictors is successful. This contains the r-squared values when iteratively adding ordered testing predictors (ordered by relative importance of the training data set).

Examples

```
#Below performs single core relative importance principal
#components regression of mpg against cyl, disp, and hp (all from the mtcars
#sample data set), optimizing for a r-squared value of 0.75.
y = mtcars$mpg[1:20]; x = mtcars[1:20,c("cyl","disp")]
pcr_object = RelimpPCR(Y = y, X = x,target_r2 = 0.75, multicore = FALSE,
remove_factors = FALSE, normalize_data = FALSE, plot_this = FALSE)
```

`RelimpPCR.predict` *Predictor Function for RelimpPCR*

Description

This function takes the trained RelimpPCR object and proceeds to perform a prediction from the best model (as defined in the documentation of RelimpPCR()).

Usage

```
RelimpPCR.predict(pcr, newdata)
```

Arguments

`pcr`	(pcr_object): The trained RelimpPCR object produced by the RelimpPCR() function.
`newdata`	(data frame): The new X value(s) you wish to draw a prediction from.

Value

pred (data frame): A data frame containing the preictions.

Examples

```
#The below function takes a trained PCR object (produced by RelimpPCR)
#and a dataframe (using the same columns that the PCR object was trained
#with) and produces a prediction.
y = mtcars$mpg[1:20]; x = mtcars[1:20,c("cyl","disp")]
pcr_object = RelimpPCR(Y = y, X = x,target_r2 = 0.75, multicore = FALSE,
                      remove_factors = FALSE, normalize_data = FALSE, plot_this = FALSE)
pred = RelimpPCR.predict(pcr_object, data.frame(mtcars$cyl, mtcars$disp))
```

`train_test_split` *A Train/Test Split Function*

Description

This function splits the data into training and testing sets.

Usage

```
train_test_split(x, y, validation_split)
```

Arguments

`x`	(data frame): A data frame of predictors.
`y`	(list/vector): A list or vector of responses.
`validation_split`	(numeric): A number between 0 and 1 that represents the proportion of the data to be used for testing.

Value

out (list): A list containing all of the below components...

$train_x (data frame): A data frame of predictors for the training set.

$train_y (list/vector): A list or vector of responses for the training set.

$test_x (data frame): A data frame of predictors for the testing set.

Index